Towards a Visual Analytics Workflow for Cybersecurity Simulations

Vit Rusnak[a] and Martin Drasar[b]
Institute of Computer Science, Masaryk University, Brno, Czechia, Czech Republic

Keywords: Cybersecurity, Attack Simulator, Visualizations, Analytical Workflow, Task Typology.

Abstract: One of the contemporary grand challenges in cybersecurity research is designing and evaluating effective attack strategies on network infrastructures performed by autonomous agents. These attackers are developed and trained in simulated environments. While the simulation environments are maturing, their support for analyzing the simulation data remains limited, mainly to inspect individual simulation runs. Extending the analytical workflow to compare multiple runs and integrating visualizations could improve the design of both attack and defense strategies. Through our work, we want to spark interest in the largely overlooked domain of visual analytics for cybersecurity simulation workflows. In this paper, we *a)* analyze the current state of the art of using visualizations in cybersecurity simulations; *b)* conceptualize the three-tier analytical workflow and identify user tasks with suggested visualizations for each tier; *c)* demonstrate the use of visualizations that augment existing CYST simulator on several real-world tasks and discuss the limitations and lessons learned.

1 INTRODUCTION

Research on attack strategies is receiving increasing attention in the cybersecurity community due to the growing scale and severity of real-world attacks. One of the leading research directions is *autonomous software agents* implementing the latest advances in machine learning and other artificial intelligence methods. These agents are expected to dominate cybersecurity warfare this decade. Their design and development require large volumes of training data and *simulation environments*.

The simulation environments enable testing and evaluating attackers' behavior in a secure virtual environment where simulations can be easily adjusted, repeated, and executed concurrently. These environments have already matured in their core features, such as configurability, scalability, and logging. However, they usually lack analytical tools or provide only limited support focused solely on a single simulation run inspection. Despite visualizations used in operational cybersecurity, network traffic anomaly detection, or malware and forensic analyses, their potential in autonomous agent research still needs to be explored.

We report on our work toward improving analytical capabilities of cybersecurity simulation researchers' through visualizations. In this paper, we address three research questions:
RQ1: What are the commonly used visualizations in cybersecurity attack simulators?
RQ2: Which are the main analytical tasks when working with the simulation data?
RQ3: How can visualizations support analysts' work?

Our contributions are, therefore, three-fold: *a)* an overview of the visualization usage in analyzing the simulation data; *b)* a conceptualization of cybersecurity simulation analytical workflow; *c)* sample use cases demonstrating two prototype visualizations to solve real-world analytical tasks.

2 RELATED WORK

In this section, we address the *RQ1*. Based on the literature review, we showcase examples of simulators used in cybersecurity research and discuss the visualizations used in such a context. In the review, we focused on virtual computer network simulators that have already been demonstrated as a cybersecurity research vehicle.

2.1 Network Simulators

Network Security Simulator – NESSi2 (Grunewald et al., 2011) is an open-source agent-based simula-

[a] https://orcid.org/0000-0003-1493-2194
[b] https://orcid.org/0000-0002-9623-1756

Rusnak, V. and Drasar, M.
Towards a Visual Analytics Workflow for Cybersecurity Simulations.
DOI: 10.5220/0011695000003417
In *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - Volume 3: IVAPP*, pages 179-186
ISBN: 978-989-758-634-7; ISSN: 2184-4321

tion environment focused on integrating and evaluating Intrusion Detection Systems (IDS). Its model allows the creation of custom network topology and defining scenarios in which the topology will be used. The simulator provides a graphical user interface, a simulation engine, and a database with results where the simulation data are stored. The user interface allows for configuring the components of the simulations and visualizing the simulation results.

Objective Modular Network Testbed in C++ – OMNeT++ (Varga, 2010) is an open-source, extensible modular framework primarily designed for network simulation. It provides a runtime environment and a domain-specific language for network description. Unlike most available simulators, it allows defining of arbitrary types of networks and communication protocols. While intended for network traffic simulations, it also offers support for executing and evaluating network attack scenarios (Michalíková, 2018; Kružík, 2018).

Real-time Immersive Network Simulation Environment for Network Security Exercises (Liljenstam et al., 2005) was designed to improve networks' readiness for possible attacks and security exercises through realistic simulation of network behavioral attacks. The networks can contain hundreds of subnets and users compared to the other simulators.

CyberBattleSim (Microsoft Defender Research Team, 2021) is an experimental platform for exploring the interaction of automated agents in a simulated network environment. The platform offers a high-level abstraction of networking and cybersecurity concepts and uses the *OpenAI Gym library* (Brockman et al., 2016) to train autonomous agents using reinforcement learning. The simulation environment uses a fixed network topology and allows the definition of stochastic defenders to implement defensive measures.

The *CYST* (Drašar et al., 2020) simulator, which addresses the shortcomings of the previous simulators, is a discrete event-based message-passing simulator enabling extensive customization of agents' actions and impacts. It validated NATO's reference architecture implementation for autonomous cybersecurity agents (Theron et al., 2020). We use it as a base environment for evaluating our visualization approaches.

2.2 Visualizations in Network Simulators

While there is broad research in cybersecurity visualizations (Damaševičius et al., 2019), the use of visualizations in cybersecurity simulators is mostly unexplored. A literature survey and examination of existing simulators revealed three visualization types predominantly used: network graphs, attack graphs, and sequence diagrams. Other common chart types (e.g., line or bar charts) are occasionally used to summarize statistical information.

Network graphs are used to model the network topology in which the simulation runs. Nodes represent individual machines in the network, and edges are the connections between them. Network graphs can also model communication (Minarik and Dymacek, 2008) or logical topology, i.e., the arrangement of elements into separate segments and virtual networks that are defined by an addressing mechanism. Edges and nodes can be provided with additional attributes to represent their state (e.g., node attacked/abused, link utilization, or unavailability). Moreover, nodes might represent different devices, such as routers, servers, or workstations. The advantage of network graphs in cybersecurity simulators is their illustrative nature. They make it easy to visualize the progress of an attack and, if the information is animated and combined with an attack timeline, illustrate its different phases. A significant disadvantage of network graphs is that their readability decreases as the number of edges increases since the resulting network often resembles hairballs for large topologies with hundreds of edges.

Attack graph visualizations represent the most common visualization types used to depict attack vectors (Yi et al., 2013). They allow identifying weak spots that an attacker can exploit. Attack graphs are frequently modeled either as modified network graphs (Homer et al., 2008) or using the UML sequence diagrams (Blaha and Rumbaugh, 2005). While the former is a network-centric representation, the latter is an attacker-centric representation that allows for modeling time dependencies and simultaneous processes, which is beneficial mainly for distributed attacks.

3 ANALYTICAL WORKFLOW

This section addresses the *RQ2* using the CYST simulator as a model environment. We first present the data produced by the simulator, followed by an introduction to the three-tiered analytical workflow and user goals. We also suggest suitable visualizations for each tier. Although other simulation tools may differ in implementation details, the task categorization is guided by the method described in (Brehmer and Munzner, 2013).

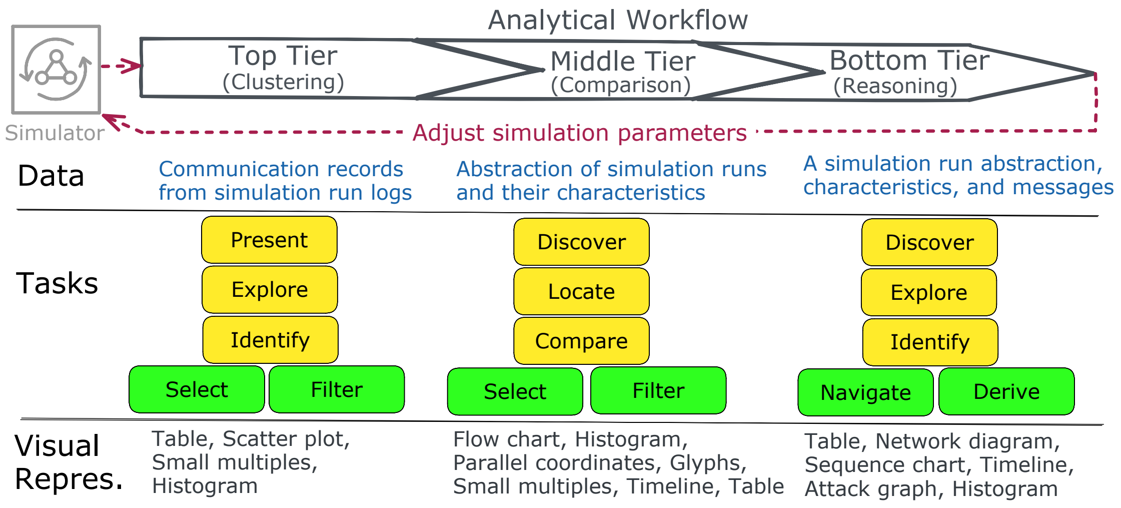

Analytical Workflow
Top Tier (Clustering)
Middle Tier (Comparison)
Bottom Tier (Reasoning)
Simulator
Adjust simulation parameters
Data
Communication records from simulation run logs
Abstraction of simulation runs and their characteristics
A simulation run abstraction, characteristics, and messages
Tasks
Present
Explore
Identify
Select
Filter
Discover
Locate
Compare
Select
Filter
Discover
Explore
Identify
Navigate
Derive
Visual Repres.
Table, Scatter plot, Small multiples, Histogram
Flow chart, Histogram, Parallel coordinates, Glyphs, Small multiples, Timeline, Table
Table, Network diagram, Sequence chart, Timeline, Attack graph, Histogram

Figure 1: Proposed formalization of the analytical workflow for cybersecurity simulations based on methodology by (Brehmer and Munzner, 2013), defined by data, tasks, and possible visual representations. The ask color indicates *why* it is performed (yellow) and *how* (green).

3.1 Simulation Data

The CYST simulator implements the Action-Intent Framework (Moskal and Yang, 2020) attack model, which contains actions that attackers (i.e., agents) choose to explore the network, gain new permissions, exploit sessions, or exfiltrate and delete data. The combination of these actions results in a so-called attack scenario. Actions are communicated to targets in the form of requests sent as messages in the simulated network. Each message may contain additional metadata processed by other components. For example, a message containing an action to perform a service scan may stochastically receive an event flag identifying it as a scan, thus simulating detection by an Intrusion Detection System (IDS). Other components may then respond appropriately based on this event flag. In our implementation, event flags for attack actions are generated with predefined probabilities, simulating some risk of detection. A message containing an event flag is considered an attack action. The generation of event flags for defensive actions is always deterministic.

The simulator run creates a communication report, i.e., the list of sent `REQUEST`s containing the attack actions, and `RESPONSE`s with the related results. There are up to 13 attributes encoded in each message:

- `type` – message type (`REQUEST` or `RESPONSE`);
- `id` – identifier (ID);
- `{src|dst}_ip` – source/destination address;
- `{src|dst}_id` – source/destination ID;
- `hop_{src|dst}_ip` – intermediate node address;
- `hop_{src|dst}_id` – intermediate node ID;
- `{src|dst}_service` – attacked service;
- `ttl` – time-to-live parameter;
- `action` – performed attack action;
- `result` – response to the attack; e.g., when the attack was successful: `SERVICE|SUCCESS`.

3.2 Analytical Workflow and Tasks

The overarching goal of cybersecurity researchers working with the simulator is to design, develop, and test (semi-)autonomous cybersecurity systems. Based on the informal discussions with domain experts participating in the CYST simulator development and with the second author as its lead architect, we formalized the analytical workflow (Figure 1) using the guidelines presented in (Brehmer and Munzner, 2013). We identified three tiers of the analytical workflow with respective tasks and proposed visualizations suitable for each tier.

3.2.1 Top Tier

At the highest tier, it is necessary to identify simulator runs with similar properties and find possible outliers that could be further compared in more detail. The fundamental problem at this tier is thus the choice of *classification criteria*. In the CYST simulator, we define them as a set of Cartesian products of message attributes and their values obtained from the simulation runs. The system should therefore support analysts in creating their classification criteria, allowing exploratory analysis and clustering. In terms of visualizations, dimensionality reduction techniques and clustering algorithms are appropriate here. Following the cluster analysis results, the analyst can focus on inspecting individual runs (see bottom tier) or compare two or more runs from the same or different clusters (see middle tier). Thus, the system should provide methods for individual and group selection of individual runs.

Related Analytical Tasks:

- *Present* and *explore* general patterns in simulation runs and their groupings based on user-defined criteria.
- *Identify* simulation runs where the attack agents were (un)successful or where the attack agents behave differently/similarly based on user-defined criteria (e.g., outliers).
- *Filter* and *select* one or more simulation runs for detailed inspection (see bottom tier) or comparison (see middle tier).

3.2.2 Middle Tier

When comparing two or more runs, the analyst's goal is to find similarities and differences between the simulation runs, based on which they can infer necessary adjustments to agent configurations. In addition to general summaries of simulation run times or message counts, this comparison aims to find suitable visual representations for all (un)successful (e.g., flow diagrams). At the same time, the interface should allow filtering and searching for messages according to their attributes.

Related Analytical Tasks:

- *Discover* and *compare* simulation runs in terms of their parameters (e.g., number of messages, runtime, count of (un)successful attack attempts)
- *Locate* critical points in simulation runs that influence the ability of agents to reach their goals.
- *Compare* the attack agent configurations for similarities and differences.
- *Filter* and *select* one simulation run for its detailed inspection (see bottom tier).

3.2.3 Bottom Tier

The lowest level follows the current practice as described in Section 2. The focus is on one simulation run and the detailed analysis of the agents' decisions leading to either successful or unsuccessful attacks. Except for statistical information such as the count of sent messages, (un)successful attacks, or their duration, the analyst request the ability to visualize attack graphs with the attack vectors and dynamically reconstruct the attack by replaying messages on the network topology.

Related Analytical Tasks:

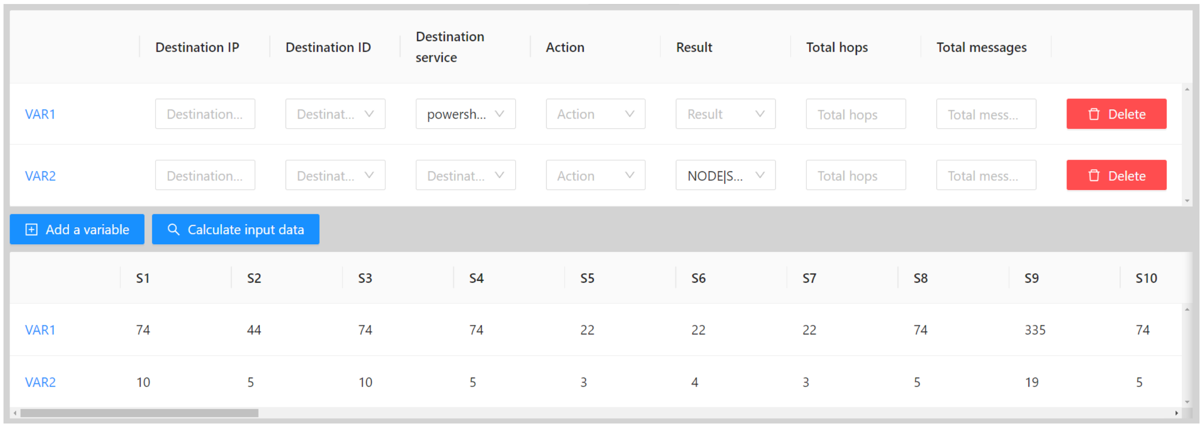
- *Discover* and *explore* communication patterns (e.g., message sequence, order of breached computers).
- *Identify* the sources of unsuccessful attacks (e.g., the wrong order of requests and loops in the agent's attack path).
- *Navigate* through (replay) a single simulation run.
- *Derive* adjustments to the attacker's model.

4 IMPLEMENTED VISUALIZATIONS

We implemented a workflow for criteria creation and cluster analysis to demonstrate the potential of using clustering visualizations in analyzing data. We developed two prototype tools for processing the CYST simulation run records. The *Clustering Analyzer* provides the initial criteria classification (top tier), and the *Scenario Player* allows to replay of a single simulation run data (bottom tier). Both tools are web applications implemented using D3.js (Bostock et al., 2011) and DruidJS (Cutura et al., 2020) libraries.

4.1 Top Tier: Clustering Analyzer

The tool parses multiple logs of the simulation runs and stores them in the database. The analysts can define their criteria and perform initial clustering to identify subsets of runs worth inspecting. The analysis process has four steps. First, the analyst loads the records of simulation runs. Next, they define filtering criteria from the message attributes. Filters are Cartesian products of message parameters that the tool uses to generate an input data table for further processing (see Figure 2). The table rows represent the dimensions, and columns correspond to individual simulation runs. The cells contain the number of messages satisfying the filtering condition (e.g., a count of successful exploits on a selected IP address). Generating a table of at least two independent variables is necessary for initiating a cluster analysis. If there are more than two dimensions, the chosen dimensionality reduction algorithm first computes a low-dimensional representation of the input table. In the final step, the analyst selects one of three dimensionality reduction methods: PCA (Dunteman, 2008), UMAP (McInnes et al., 2018), or t-SNE (van der



	Destination IP	Destination ID	Destination service	Action	Result	Total hops	Total messages	
VAR1	Destination...	Destinat...	powersh...	Action	Result	Total hops	Total mess...	Delete
VAR2	Destination...	Destinat...	Destinat...	Action	NODE|S...	Total hops	Total mess...	Delete

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
VAR1	74	44	74	74	22	22	22	74	335	74
VAR2	10	5	10	5	3	4	3	5	19	5

Figure 2: Cartesian filtering (top) and clustering criteria input table (bottom). Generated input table shows only two dimensions (VAR1, VAR2 defined by the filters above.

Maaten and Hinton, 2008). The hierarchical clustering algorithm (Murtagh, 2011), applied to the result, splits the data into clusters and plots them in a two-dimensional scatter plot based on the selected dimensionality reduction method. Given the exploratory nature of the analysis, we implemented both linear (PCA) and non-linear (UMAP and t-SNE) dimensionality reduction algorithms. The parameters of the latter two are configurable. On mouse over, the pop-up shows the simulation run identifier. The plotted dots represent clustered simulation runs.

4.2 Bottom Tier: Scenario Player

In the bottom tier, the analyst explores the behavior of an attacker in a single simulation run. We have created an interactive scenario player to make it easier for them to identify points of interest. The tool loads a dataset of a single simulation run and the network infrastructure configuration. The topology is rendered as a network graph with automated node positioning. The analyst can reposition them if needed.

Initially, the nodes are color-coded to distinguish between attacking and target network nodes. Initially, all nodes are colored green. Only already-known attackers are red. If nodes in the network are compromised throughout the attack scenario, their color changes from green to amber. Such encoding enables identifying the moment of an attack when the compromise happened, the sequence of actions leading to it, and the possible impact on other nodes in the network. The configuration options (Figure 3, top-right) enable to toggle of four analytic visualizations: *Node activity* helps identify whether a node is predominantly sending or receiving messages; *Attack progress*, adds node numbers indicating the order of successful attacks; *Successful attacks*, enables a tooltip on hover displaying successful attack actions for a given node; and *All successful attacks*, displays a summary of all successful attack actions for a given scenario.

The analyst can also inspect messages by invoking an on-mouseover tooltip with message attributes, seeking a simulation run, or adjusting a replay speed (i.e., the ratio of simulated time units to seconds). Requests and responses are color-coded for easier differentiation – green for requests and orange for responses.

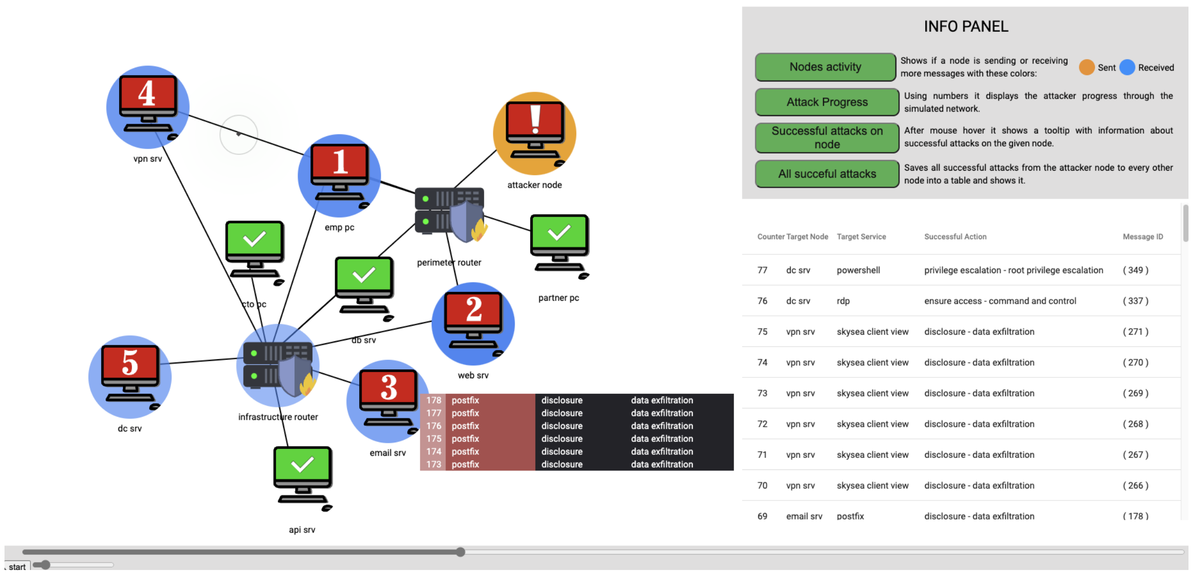
5 USAGE SCENARIOS

To demonstrate the usefulness of the implemented prototypes on the analytical workflow (*RQ3*), we present several use cases created by the CYST simulator developers. The use cases utilize the tools to achieve real examples of analytical goals.

5.1 Clustering Analyzer Use Cases

Each simulation was run 50 times with the same attacker configuration and topologies described in (Drašar et al., 2020). Given the limited scope, only selected visualizations of the results are presented.

Case 1: Identify different attackers' tactics. The simulation was performed on the topology ***employee***. The analyst explored different tactics for service attacks. Six variables were defined as a cartesian product of *servers* through which the attack passed (`web_srv`, `db_srv` and `dc_srv`) and the *response* values (`SERVICE|SUCCESS` and `SERVICE|FAILURE`). t-SNE cluster analysis identified three clusters, each representing one attack tactic utilizing a specific server (see Figure 4).



Shows if a node is sending or receiving more messages with these colors: ● Sent ● Received

Using numbers it displays the attacker progress through the simulated network.

After mouse hover it shows a tooltip with information about successful attacks on the given node.

Saves all successful attacks from the attacker node to every other node into a table and shows it.

Counter	Target Node	Target Service	Successful Action	Message ID
77	dc srv	powershell	privilege escalation - root privilege escalation	(349)
76	dc srv	rdp	ensure access - command and control	(337)
75	vpn srv	skysea client view	disclosure - data exfiltration	(271)
74	vpn srv	skysea client view	disclosure - data exfiltration	(270)
73	vpn srv	skysea client view	disclosure - data exfiltration	(269)
72	vpn srv	skysea client view	disclosure - data exfiltration	(268)
71	vpn srv	skysea client view	disclosure - data exfiltration	(267)
70	vpn srv	skysea client view	disclosure - data exfiltration	(266)
69	email srv	postfix	disclosure - data exfiltration	(178)

178	postfix	disclosure	data exfiltration
177	postfix	disclosure	data exfiltration
176	postfix	disclosure	data exfiltration
175	postfix	disclosure	data exfiltration
174	postfix	disclosure	data exfiltration
173	postfix	disclosure	data exfiltration

Figure 3: Scenario Player dynamically visualizes a single simulation run. Red nodes are attacking. Numbered nodes are compromised, green yet uncompromised. The colored circle represents whether a node sends (amber) or receives (blue) more messages. Its opacity represents the message count relative to other nodes. Logs related to a selected node show in a pop-up (center-bottom), and all the successful actions are listed on the right.

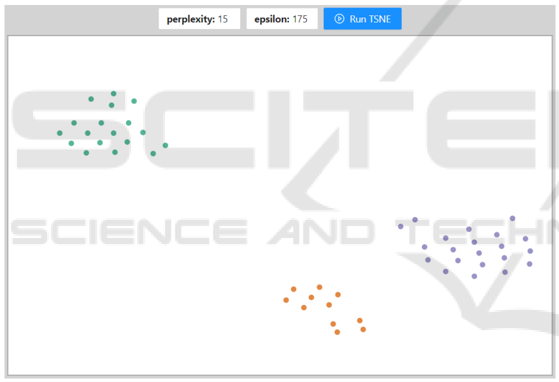


Figure 4: Result of cluster analysis aimed at showing attackers with different tactics (t-SNE).

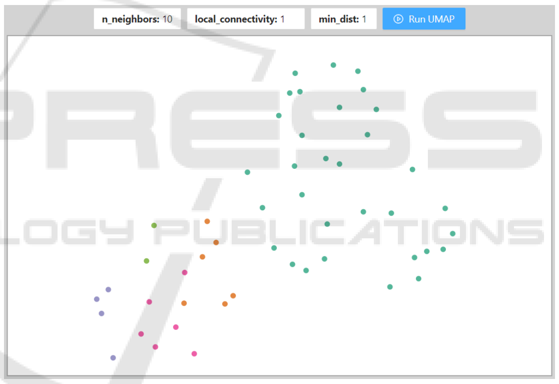


Figure 5: Result of cluster analysis aimed at identification of attackers able to ignore unimportant targets. (UMAP).

Case 2: Identify attackers' ability to ignore unimportant targets. The simulation was performed on the topology ***cto***. The analyst defined five variables to indicate requests on servers unimportant for a successful attack: `emp_pc`, `email_srv`, `api_srv`, `vpn_srv`, and `web_srv`. The resulting UMAP algorithm showed (Figure 5) a more extensive cluster (green dots). Upon inspection of the runs, the analyst concluded that the attacker quickly recognized unimportant targets (i.e., terminated the communication with them) and ignored them in consecutive steps.

Case 3: Explore attackers prone to select inappropriate actions. The simulation was run on the topology ***cto***. Five variables combined servers from case 2 with the response value `SERVICE|SUCCESS`. Ideally, this value should occur only once on each service. The PCA method (Figure 6) showed multiple clusters worth exploring. A closer look revealed clusters where simulation runs have higher variable counts, i.e., the attacker's algorithm chose inappropriate actions of an exploratory nature rather than those that would allow a successful attack to progress.

5.2 Scenario Player Use Cases

The benefit of the single scenario run replays lies in observing the attack unfolding, thus enabling identification and analysis of their time-related aspects. Further, we briefly discuss two use cases the Scenario Player revealed.

Case 4: Identify the general pattern and inefficiencies of the attack. As the nodes get progres-

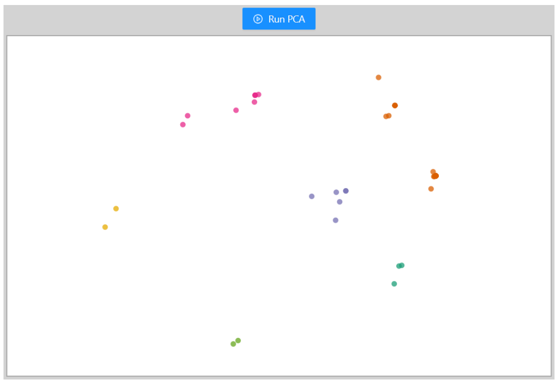


Figure 6: Result of cluster analysis aiming to identify attackers prone to select inappropriate actions (PCA).

sively numbered to identify compromised hosts, the analyst can identify preferred or ignored targets or networks and observe the attack's path in the context of the whole infrastructure. Large amounts of request/response pairs exchanged between pairs of network nodes indicate an unnecessarily extensive volume of network traffic, thus pointing to inefficient attack strategies. Further exploration of target services, used attacks, and other variables can help identify weaknesses in attack strategies.

Case 5: Explore deadlocks. When the analyst sees that there is not any newly compromised node for an extended period, the attack is stalling. In such cases, attackers usually aim at invalid targets or make wrong attack strategy decisions. Scenario replay can pinpoint the causes of this deadlock and help the analyst derive possible changes in attacker configurations to avoid it.

6 DISCUSSION

Further, we discuss the limitations and validity threats that we are aware followed by the presentation of lessons learned from our ongoing work.

6.1 Validity Threats and Limitations of Our Approach

We identified three main threats to the validity of our approach: relevance to the research community, minimal opportunities to validate the outputs, and the low maturity level of the prototypes.

The starting point for the proposed formalization of the analytical workflow and the classification of actions in each tier is the CYST simulator model environment (see Section 3). However, the resulting description of the analytical workflow, task categories, and examples of visual representations generally apply to other simulators despite their specificities in log and report formats.

The specific focus on the problem of analyzing data from cybersecurity simulators has determined a significant limitation regarding the validation of our results. First and foremost, there need to be more domain experts (i.e., cybersecurity simulation researchers and analysts). The issue of defensive and offensive strategies is still being investigated by a relatively small group of cybersecurity researchers worldwide. Thus, naturally, in our work, we also encountered the problem of finding other domain experts outside the team developing the CYST simulator. Therefore, it will be helpful to subject the results of our work to a thorough discussion, especially among domain experts.

Lastly, the presented prototypes were used to demonstrate the initial usability of the proposed workflow. They suffer from disconnectedness and low maturity. We also intentionally bypassed the prototyping of middle-tier visualizations that will require even more careful analysis and workflow refinement.

6.2 Lessons Learned

Unexplored territory: Although the use of visualizations in cybersecurity practice is now commonplace and is contributed to by the community around the *IEEE Symposium on Visualization for Cyber Security (VizSec)*, the main directions of visualization usage are mainly for network traffic data or supporting forensic analysis. Cybersecurity simulations are still an unexplored area that provides many research opportunities.

Common visualizations over domain-specific ones: Since analysts are usually not experts in visualizations, it is advisable to prefer commonly known visual representations, as they eliminate the risk of misinterpretation. The only domain-specific case is the attack graph, which is widely used in the cybersecurity community.

Clustering analyzer generalization: The proposed tool can also be used for data analysis in other domains where it is appropriate to derive abstract criteria based on the combination of the input data attributes. Verbal feedback from the CYST simulator developers indicated that even the generated table with input criteria was sufficient for the initial orientation in the data. The tool can also allow comparing different dimension reduction methods, e.g., depending on the input data. It also can be easily extensible to other methods included in the DruidJS library (Cutura et al., 2020).

7 CONCLUSION

In this paper, we have tackled using visualizations in an analytical workflow of cybersecurity simulator data. To our knowledge, this paper is the first attempt to conceptualize the work in this domain. We described the state-of-the-art network simulators and commonly used visualizations in the context of cybersecurity simulation research of autonomous agents (*RQ1*). ext, we mapped the analysts' tasks and formalized the analyst workflow (*RQ2*) based on the CYST simulator as a model environment. Finally, we presented two prototype tools – Clustering Analyzer and Scenario Player – and five use cases to demonstrate their suitability to deal with several real-world analytical goals (*RQ3*). We also discussed limitations and provided lessons learned.

Further, we plan to merge the tools into a single data analytics toolkit integrated seamlessly into a CYST simulator workflow. It will also include integrating novel visualizations and advancing the middle tier according to the defined three-tier analytical workflow.

ACKLNOWLEDGEMENTS

The research was supported by ERDF "CyberSecurity, CyberCrime and Critical Information Infrastructures Center of Excellence" (No. CZ.02.1.01/0.0/0.0/16_019/0000822).

REFERENCES

Blaha, M. R. and Rumbaugh, J. E. (2005). *Object-oriented Modeling and Design with UML, 2nd Ed.* Pearson Ed.

Bostock, M., Ogievetsky, V., and Heer, J. (2011). D^3 Data-Driven Documents. *IEEE TVCG*, 17(12):2301–2309.

Brehmer, M. and Munzner, T. (2013). A Multi-Level Typology of Abstract Visualization Tasks. *IEEE TVCG*, 19(12):2376–2385.

Brockman, G., Cheung, V., Pettersson, L., et al. (2016). OpenAI Gym. arXiv:1606.01540.

Cutura, R., Kralj, C., and Sedlmair, M. (2020). DRUIDJS – A JavaScript Library for Dimensionality Reduction. In *IEEE VIS 2020*, pages 111–115. IEEE.

Damaševičius, R., Toldinas, J., et al. (2019). Visual Analytics for Cyber Security Domain: State-of-the-Art and Challenges. In *Information and Software Technologies*, volume 1078, pages 256–270. Springer Int. Publ., Cham.

Drašar, M., Moskal, S., et al. (2020). Session-level Adversary Intent-Driven Cyberattack Simulator. In *Proc. of the IEEE/ACM 24th Int. Symp. on Distrib. Simulation and Real Time Applications (DS-RT'20)*, pages 7–15. IEEE.

Dunteman, G. H. (2008). *Principal Components Analysis*. Number 69 in Quantitative Applications in the Social Sciences. Sage Publishing, Newbury Park, CA, USA.

Grunewald, D., Lützenberger, M., Chinnow, J., et al. (2011). Agent-Based Network Security Simulation. In *The 10th Int. Conf. on Autonomous Agents and Multiagent Systems – Vol. 3*, AAMAS '11, page 1325–1326, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Homer, J., Varikuti, A., et al. (2008). Improving Attack Graph Visualization through Data Reduction and Attack Grouping. In Goodall, J. R., Conti, G., and Ma, K.-L., editors, *Visualization for Computer Security*, volume 5210 of *LNCS*, pages 68–79. Springer Berlin Heidelberg.

Kružík, M. (2018). Adaptive Network Attacks. Master's thesis, Masaryk University, Brno, Czechia.

Liljenstam, M., Liu, J., et al. (2005). RINSE: The Real-Time Immersive Network Simulation Environment for Network Security Exercises. In *Workshop on Principles of Adv. and Distrib. Simul. (PADS'05)*, pages 119–128. IEEE.

McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv:1802.03426.

Michalíková, M. (2018). Implementation of Methods for Attack Detection in Software Simulator. Bachelor's thesis, Masaryk University, Brno, Czechia.

Microsoft Defender Research Team (2021). CyberBattleSim. github.com/microsoft/CyberBattleSim.

Minarik, P. and Dymacek, T. (2008). NetFlow Data Visualization Based on Graphs. In *Vis. for Comp. Sec.*, volume 5210 of *LNCS*, pages 144–151. Springer Berlin Heidelberg.

Moskal, S. and Yang, S. J. (2020). Cyberattack Action-Intent-Framework for Mapping Intrusion Observables. arXiv:2002.07838.

Murtagh, F. (2011). Hierarchical Clustering. In *International Encyclopedia of Statistical Science*, pages 633–635. Springer Berlin Heidelberg, Berlin, Heidelberg.

Theron, P., Kott, A., et al. (2020). *Reference Architecture of an Autonomous Agent for Cyber Defense of Complex Military Systems*, pages 1–21. Springer Int. Publ., Cham.

van der Maaten, L. and Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(11):2579–2605.

Varga, A. (2010). OMNeT++. In *Modeling and Tools for Network Simulation*, pages 35–59. Springer Berlin Heidelberg.

Yi, S., Peng, Y., et al. (2013). Overview on Attack Graph Generation and Visualization Technology. In *2013 Int. Conf. on Anti-Counterfeiting, Security and Identification (ASID)*, pages 1–6. IEEE.